

รายงานวิชาปฏิบัติการ

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

รหัสวิชา ____242-301____ ตอน ____01____ วัน ____อังคาร____

รหัสหัวข้อปฏิบัติการ ____2HA05____

ชื่อหัวข้อปฏิบัติการ ____Introduction to FPGA design using Verilog____

วันที่ลงปฏิบัติการ ____17 ตุลาคม 2560____

อาจารย์ผู้สอน ____อาจารย์คมสันต์ กาญจนสิทธิ์____

กลุ่มที่ ____8____

ผู้จัดทำรายงานชื่อ ____นายปณิธาน ดวงขวัญ____ รหัส ____5735512036____

ผู้ร่วมงาน ชื่อ ____นางสาวรัตนพร ไทยเกิด____ รหัส ____5735512047____

ชื่อ ____รหัส ____

สำหรับเจ้าหน้าที่

วันที่ตรวจรับ ____

ลงชื่อ ____



การทดลองที่ 3HA05

Introduction to FPGA design using Verilog

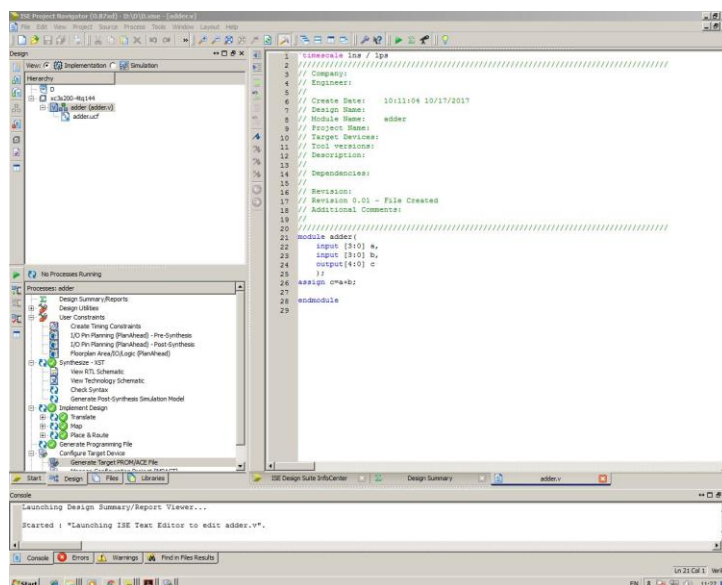
เครื่องมือและอุปกรณ์

1. โปรแกรม Xilinx
2. บอร์ด FPGA รุ่น XC3S200-TQ144C
3. คอมพิวเตอร์

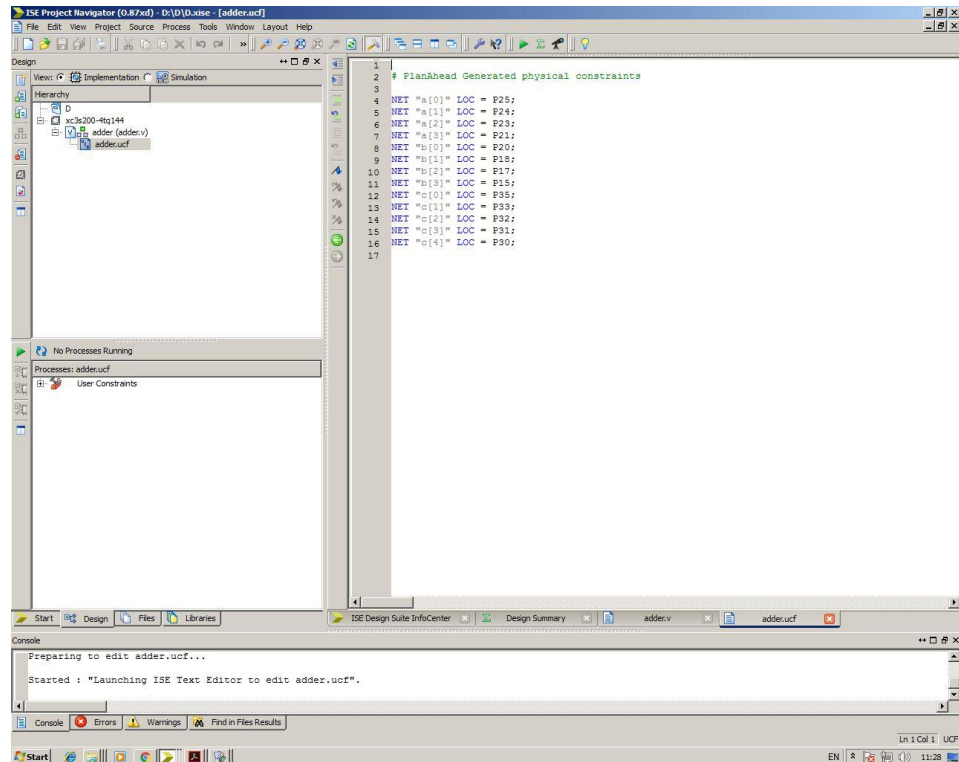
การทดลอง

1. วงจร Adder

- a. เข้าโปรแกรม Xilinx
- b. New project พร้อมกำหนดคุณสมบัติตามที่อาจารย์ให้
- c. คลิกขวาที่ตัว Project ที่เราสร้างขึ้นมาจากซ้ายมือและทำการกำหนดตัวแปรดังนี้
 - i. A เป็น input [3:0]
 - ii. B เป็น input [3:0]
 - iii. C เป็น output [4:0]
- d. ทำการกด Finish และสามารถเขียนโปรแกรมได้
- e. ใส่ code ดังนี้



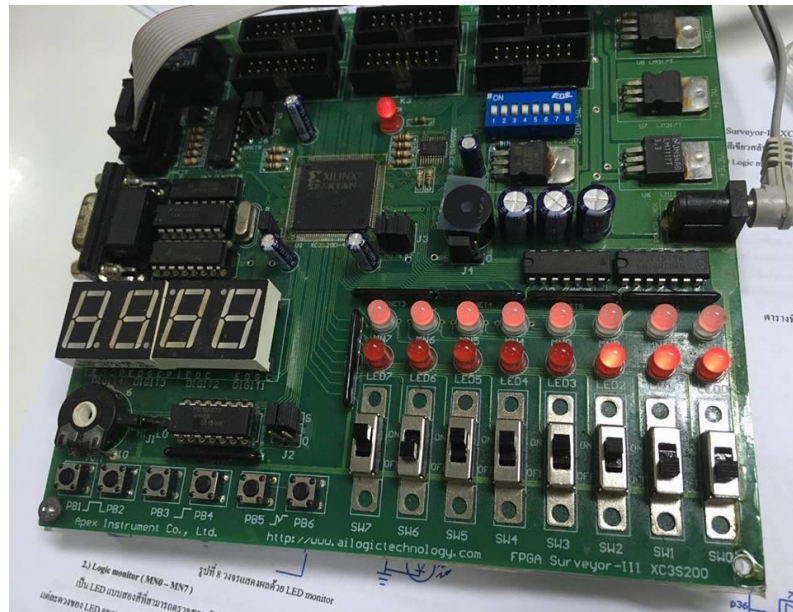
- f. ไปกำหนดขา input ให้ตรงกับบอร์ดของเรา โดยสามารถกำหนดให้ตรงโดยการดูจากบอร์ดของเรา โดยไปกำหนดที่ Source > Source for > Synthesis /Implementation และ Process > User Constraints > Assign Package Pins ดังรูป



- g. โปรแกรมลงสู่บอร์ด FPGA โดย
- i. Process > Implement Programming File เพื่อทำการ Implement วงจรที่ได้ออกแบบไว้
 - ii. Process > Generate Programming File
 - iii. Process > Generate Programming File > Configure Device(IMPACT) โดยที่ใช้จะมีนามสกุลเป็น (.bit)
 - iv. หลังจากนั้นโปรแกรมจะ Build ลงบอร์ด FPGA



สรุปผลการทดลองวงจร Adder



จากบอร์ดเราได้กำหนดเป็น Sw0 – Sw3 ของตัวแปร a / Sw4 – Sw7 ของตัวแปร b โดยจะเป็น input และ output จะเป็น LED0 – LED7 เป็นของตัวแปร c โดยเราสามารถสับสวิตช์ได้และหลังจากนั้นตัวของโปรแกรมจะทำการ บวกค่ากันและไปแสดงผลบนแผง LED นั้นเอง

2. วงจร Decoder

- a. เข้าโปรแกรม Xilinx
- b. ทำเหมือนกับ a-c ในวงจร Decoder แต่เปลี่ยนกำหนดค่าตัวแปรดังนี้
 - i. I เป็น input [2:0]
 - ii. O เป็น output [7:0]
- c. ทำการกด Finish



d. หลังจากนั้นให้เขียน code ดังนี้

The screenshot shows the ISE Project Navigator interface. The left pane displays the project hierarchy with 'decoder.v' and 'decoder.ucf' selected. The main editor shows the Verilog code for the 'decoder' module. The code defines an 8-to-1 decoder with 3 inputs and 8 outputs. The output logic is implemented using a case statement. The console window at the bottom shows the 'PlanAhead' process starting and preparing the launch script.

```

1 timescale 1ns / 1ps
2
3 //////////////////////////////////////////////////
4 // Company:
5 // Engineer:
6 // Create Date: 11:16:32 10/18/2016
7 // Design Name:
8 // Module Name: decoder
9 // Project Name:
10 // Target Device:
11 // Tool versions:
12 // Description:
13 // Dependencies:
14 //
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////
21 module decoder(
22     input [2:0] i,
23     output [7:0] o
24 );
25 //////////////////////////////////////////////////
26 reg [7:0] so;
27 assign o = so;
28 always @*
29     case (i)
30         3'b000 : so <= 8'b00000001;
31         3'b001 : so <= 8'b00000010;
32         3'b010 : so <= 8'b00000100;
33         3'b011 : so <= 8'b00001000;
34         3'b100 : so <= 8'b00010000;
35         3'b101 : so <= 8'b00100000;
36         3'b110 : so <= 8'b01000000;
37         3'b111 : so <= 8'b10000000;
38         default: so <= 8'b00000000;
39     endcase
40 //////////////////////////////////////////////////
41 endmodule
42
43

```

Console Output:

```

Started: "I/O Pin Planning (PlanAhead) - Pre-Synthesis".
Preparing PlanAhead launch script...
PlanAhead started. PlanAhead output can be found in C:/Users/comeng/Desktop/lab5/lab5/2/planAhead_run_1/planAhead_run.log

```

e. หลังจากนั้นทำการกำหนดขา input และ output ให้ตรงกับบอร์ด

The screenshot shows the ISE Project Navigator interface. The left pane displays the project hierarchy with 'decoder.ucf' selected. The main editor shows the configuration file for the decoder module, where each input and output pin is assigned to a specific physical pin on the board. The console window at the bottom shows the 'PlanAhead' process starting and preparing the launch script.

```

1
2
3 NET "i[0]" LOC = P259;
4 NET "i[1]" LOC = P249;
5 NET "i[2]" LOC = P239;
6 NET "o[0]" LOC = P359;
7 NET "o[1]" LOC = P349;
8 NET "o[2]" LOC = P339;
9 NET "o[3]" LOC = P329;
10 NET "o[4]" LOC = P319;
11 NET "o[5]" LOC = P309;
12 NET "o[6]" LOC = P279;
13 NET "o[7]" LOC = P269;
14

```

Console Output:

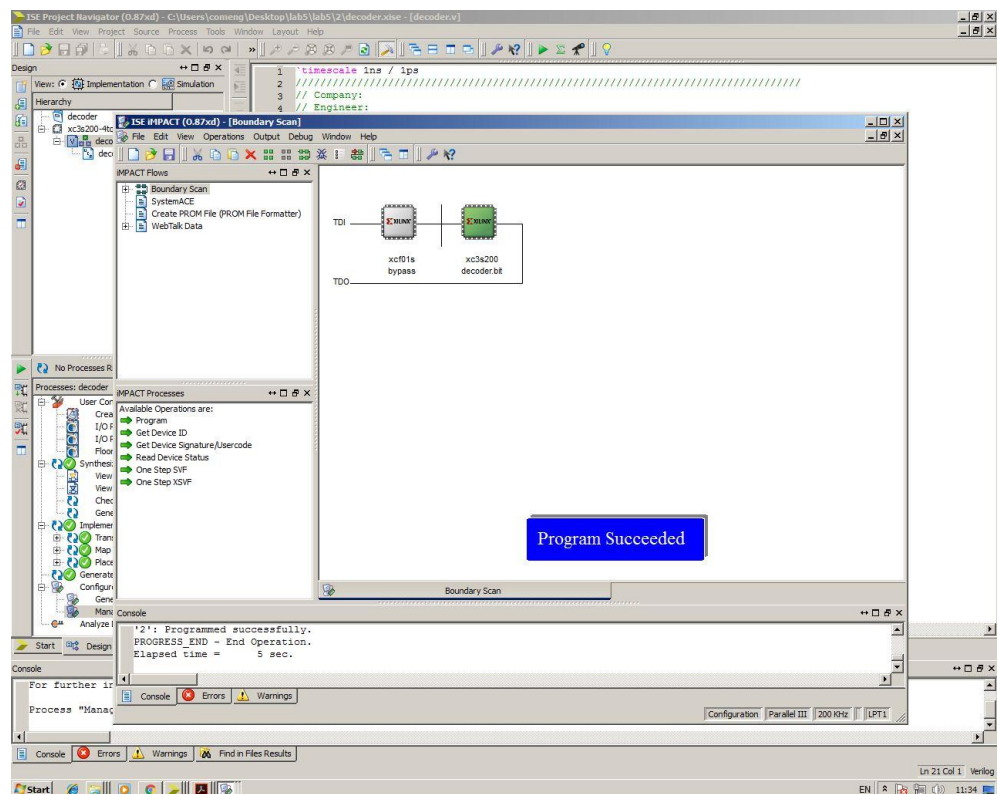
```

Preparing PlanAhead launch script...
PlanAhead started. PlanAhead output can be found in C:/Users/comeng/Desktop/lab5/lab5/2/planAhead_run_1/planAhead_run.log
Preparing to edit decoder.ucf...

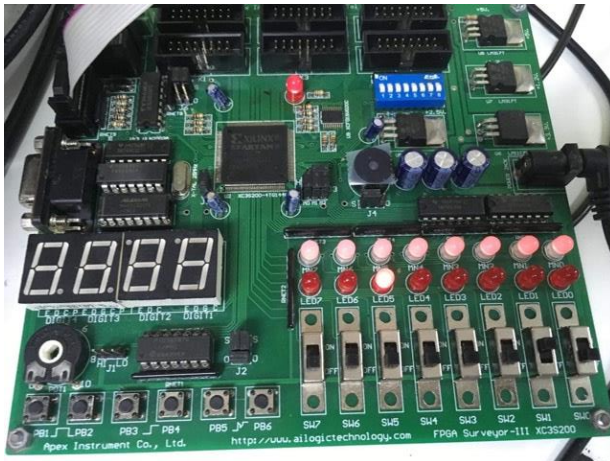
```



f. หลังจากนั้นทำเหมือนขั้นตอน g ในวงจร Adder



สรุปผลการทดลองวงจร Decoder



กำหนดให้ sw0 – sw2 เป็น input คือตัวแปร I
และ LED0 – LED7 ให้เป็น output ของตัวแปร
o โดยเราสามารถสับสวิตช์ได้ตามที่เราต้องการ
โดยจะแปลงจากโค้ดที่เราเขียนไปโดยมีเงื่อนไข
ดังนี้

000 > 00000001 / 001 > 00000010 / 010
> 00000100 / 011 > 00001000 / 100 >
00010000 / 101 > 00100000 / 110 >
01000000 / 111 > 10000000

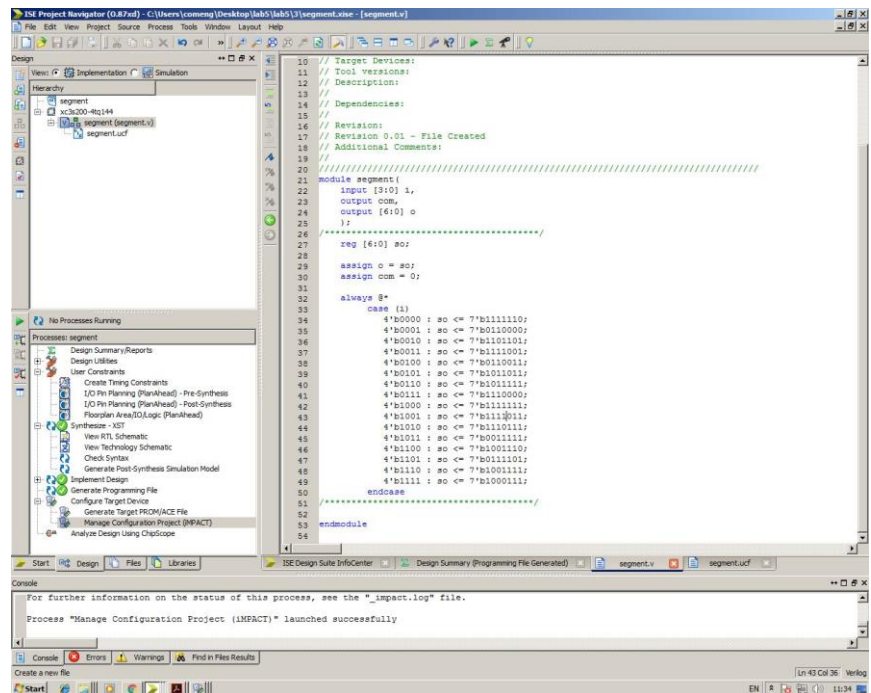


3. วงจร Segment

a. ทำตามขั้นตอนเหมือนกับทุกวงจรข้อ a – c โดยมีการกำหนดตัวแปรใหม่ดังนี้

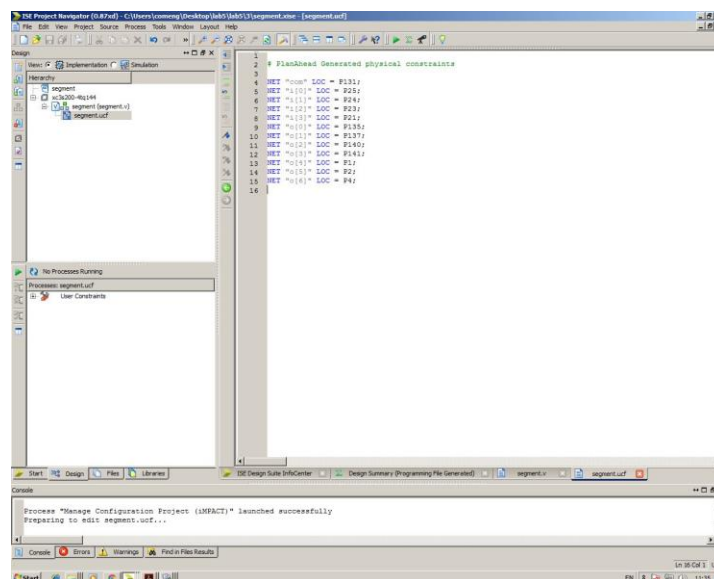
- i. I เป็น input [3:0]
- ii. O เป็น output [6:0]
- iii. Com เป็น output [0:0]

b. เขียนโค้ดดังนี้



```
10 // Target Devices:
11 // Tool versions:
12 // Description:
13 // Dependencies:
14 //
15 // Revision:
16 // Revision 0.01 - File Created
17 // Additional Comments:
18 //
19 //
20 module segment(
21     input [3:0] I,
22     output [6:0] O,
23     output [0:0] Com,
24 );
25 //
26 reg [6:0] so;
27
28 assign o = so;
29 assign com = 0;
30
31 always @*
32 case (I)
33 4'b0000 : so <= 7'b11111110;
34 4'b0001 : so <= 7'b01110000;
35 4'b0010 : so <= 7'b01101101;
36 4'b0011 : so <= 7'b11111001;
37 4'b0100 : so <= 7'b01110011;
38 4'b0101 : so <= 7'b01011011;
39 4'b0110 : so <= 7'b10111111;
40 4'b0111 : so <= 7'b11110000;
41 4'b1000 : so <= 7'b11111111;
42 4'b1001 : so <= 7'b11111011;
43 4'b1010 : so <= 7'b11101111;
44 4'b1011 : so <= 7'b00011111;
45 4'b1100 : so <= 7'b10011110;
46 4'b1101 : so <= 7'b01111101;
47 4'b1110 : so <= 7'b10011111;
48 4'b1111 : so <= 7'b10001111;
49
50 endcase
51
52 endmodule
```

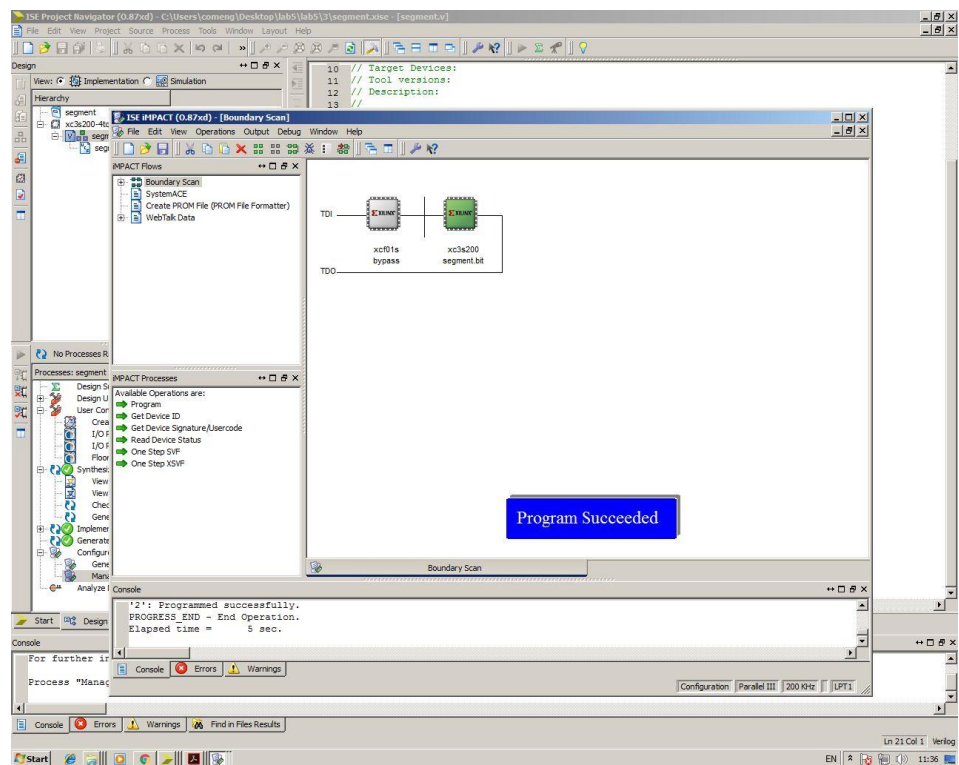
c. หลังจากนั้นให้กำหนดขาให้ตรงตามบอร์ด



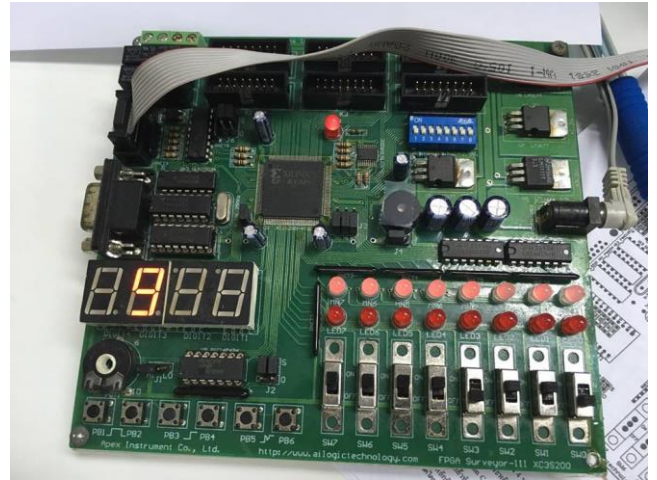
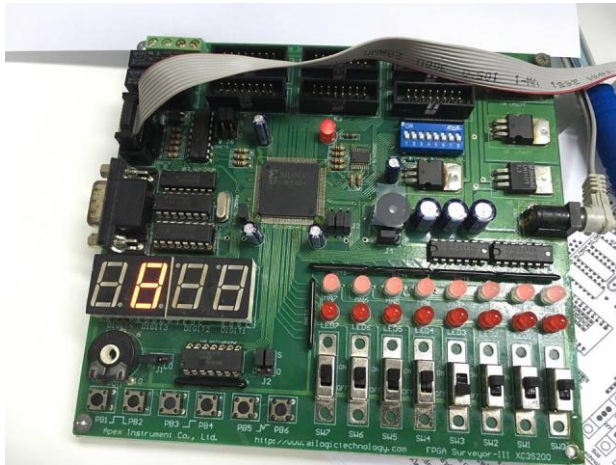
```
1 # Flashhead Generated physical constraints
2
3 #
4 # I[0:3] LOC = P131;
5 # I[0] LOC = P25;
6 # I[1] LOC = P24;
7 # I[2] LOC = P23;
8 # I[3] LOC = P22;
9 # O[0:6] LOC = P135;
10 # O[0] LOC = P137;
11 # O[1] LOC = P140;
12 # O[2] LOC = P141;
13 # O[3] LOC = P2;
14 # O[4] LOC = P2;
15 # O[5] LOC = P4;
16 # O[6] LOC = P4;
```



d. ทำการ build ลงบอร์ด FPGA โดยทำตามขั้นตอนเหมือนกับทุกวงจร



สรุปผลการทดลองวงจร 7-Segment Decoder



เราทำการกำหนด input เป็น sw0 – sw3 โดยเป็นตัวแปร 1 ส่วนของ output จะเป็น 7-segment โดยในส่วนของ 7-segment นั้นจะมีค่า 0-6 โดยจะเป็นขา a-g นั้นเองโดยจะสังเกตว่าเราจะทำการแปลง input โดยแสดงผลบน 7-segment นั้นเราจะเห็นว่ามันจะแปลงตามค่าตารางความจริงนั่นเอง

Decimal Digit	Input lines				Output lines							Display pattern
	A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	0	1	1	1	1	1	1	0	0
1	0	0	0	1	0	1	1	0	0	0	0	1
2	0	0	1	0	1	1	0	1	1	0	1	2
3	0	0	1	1	1	1	1	1	0	0	1	3
4	0	1	0	0	0	1	1	0	0	1	1	4
5	0	1	0	1	1	0	1	1	0	1	1	5
6	0	1	1	0	1	0	1	1	1	1	1	6
7	0	1	1	1	1	1	1	0	0	0	0	7
8	1	0	0	0	1	1	1	1	1	1	1	8
9	1	0	0	1	1	1	1	1	0	1	1	9

Credit : <https://www.electrical4u.com/bcd-to-seven-segment-decoder/>

